Random Comparisons of Performance Between the Cray XT3 and Cray XT4 at ORNL

(formerly Cray XT3/XT4 Performance Analysis)

Patrick H. Worley
Oak Ridge National Laboratory

The First Annual Cray Technical Workshop - USA
February 26-28, 2007
Gaylord's Opryland Hotel
Nashville, Tennessee

Acknowledgements

- Research sponsored by the Office of Mathematical, Information, and Computational Sciences, Office of Science, U.S. Department of Energy under Contract No. DE-AC05-00OR22725 with UT-Battelle, LLC.

- Oak Ridge National Laboratory is managed by UT-Battelle, LLC for the United States Department of Energy under Contract No. DE-AC05-00OR22725.

Cray XT3 vs Cray XT4

1. XT4 memory is 60% faster.
2. MPI bandwidth on the XT4 is twice that on the XT3.
3. XT4 uses a new programming environment, with new performance-related environment variables.

Questions:

What is the impact of these changes? Do users need to change how they use the machine, retuning the performance of their codes? What performance differences should they expect when running on the XT4 as compared to the XT3? Are the performance differences significant enough that mixing XT3 and XT4 nodes in a single job on the hybrid system will negate the advantage of running on XT4 nodes?

Methodology

Comparing XT3 and XT4 in areas of

1. Kernel and application performance
2. Performance sensitivities:
   a) single core (-SN) vs. dual core (-VN)
   b) compiler optimization options
   c) runtime options, e.g. -small_pages and environment variables
   d) MPI protocols and collectives
3. Performance impact of new environment variables

Outline of Talk

(guaranteed not to exceed)

Verification of Claims

1. MPI tests
2. PSTSWM (Parallel Spectral Transform Shallow Water Model)
   a) Serial performance and performance sensitivities

Measure of Impacts

3. POP (Parallel Ocean Program)
4. CAM (Community Atmosphere Model)
5. Cubed Sphere Finite Volume Dynamical Core

Sensitivities

6. * PSTSWM (Parallel Spectral Transform Shallow Water Model)
   b) Parallel performance and performance sensitivities

* See CUG talk in Seattle

COMMTEST Benchmark

- COMMTEST is a suite of codes that measure the performance of MPI interprocessor communication. In particular, COMMTEST evaluates the impact of communication protocol, packet size, and total message length in a number of "common usage" scenarios. (However, it does not include persistent MPI point-to-point commands among the protocols examined.)
- Compiled with -fast and run with -small_pages (the most common options used in my application experiments) and with

  `setenv MPICH_RANK_REORDER_METHOD 1`

  so that processes have the expected order in the experiments.

COMMTEST Experiments

i-j

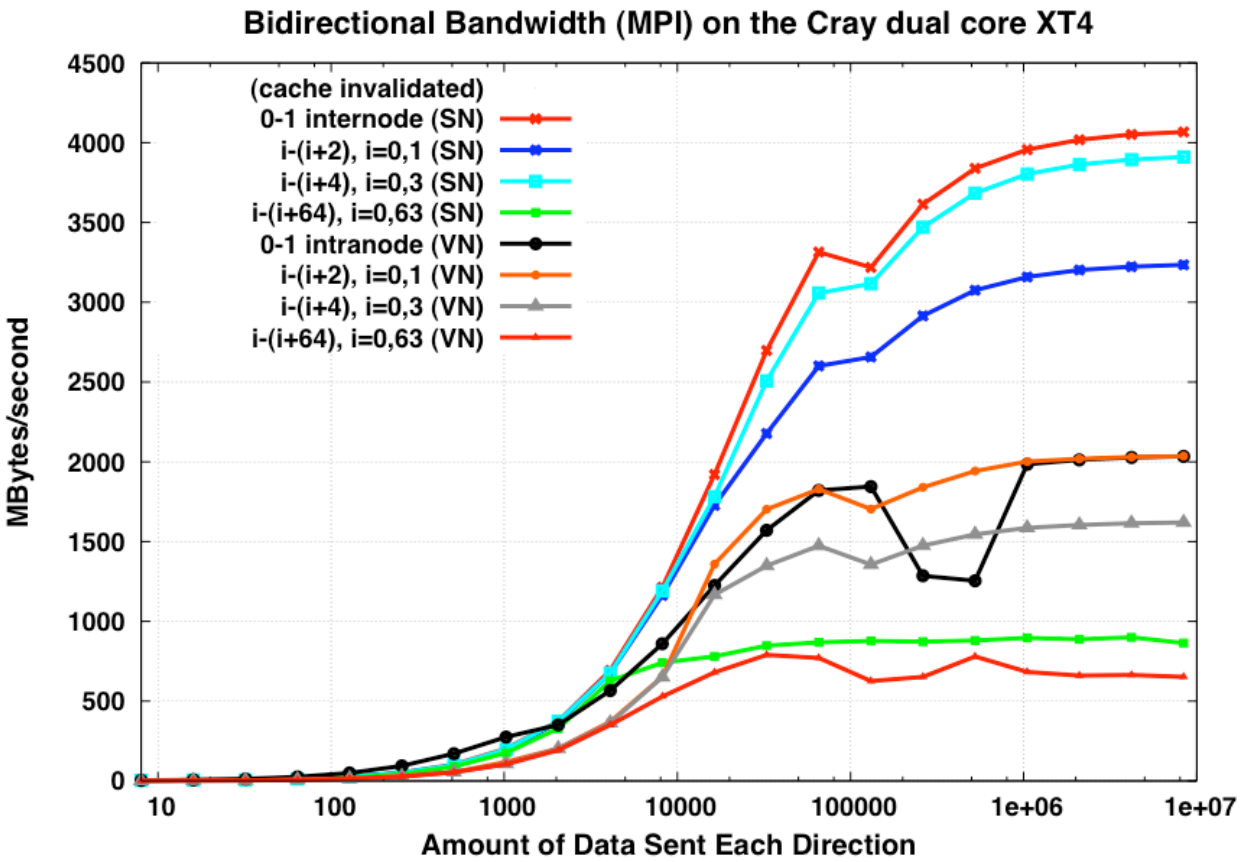
processor i swaps data with processor j. Depending on i and j, this can be within a node or between nodes.

i-(i+j); i=1,…,n; n<j

n processor pairs (i,i+j) swap data simultaneously. Depending on j, this will be within a node or between nodes (or both). Minimum per pair performance is reported.

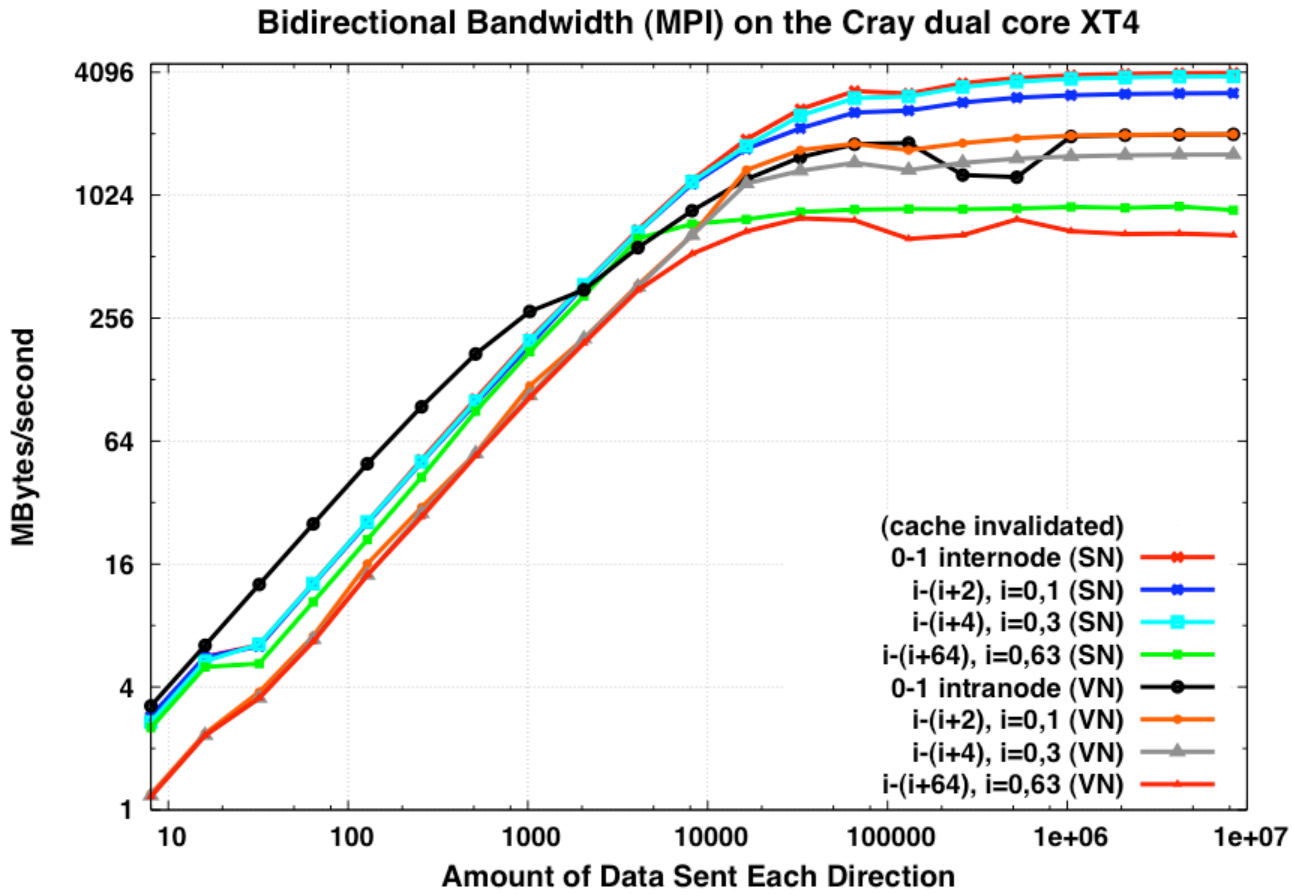
Note: experiments were not run in dedicated mode. Nodes were usually, but not always, physically “contiguous”, but part of torus used was not controlled as part of the experiment.

SWAP Benchmark on XT4



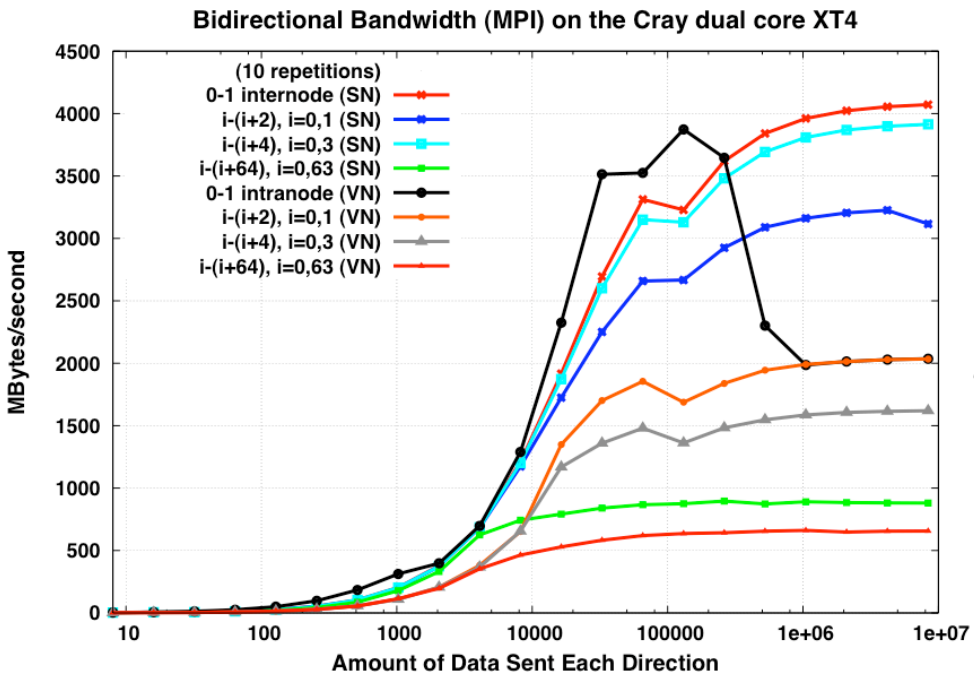
Comparing per processor pair performance of SWAP for different communication patterns. Contention for internode bandwidth limits the single pair bandwidth for the simultaneous exchange VN experiments. Note that the counterintuitive 4-node and 8-node simultaneous exchange results are reproducible.

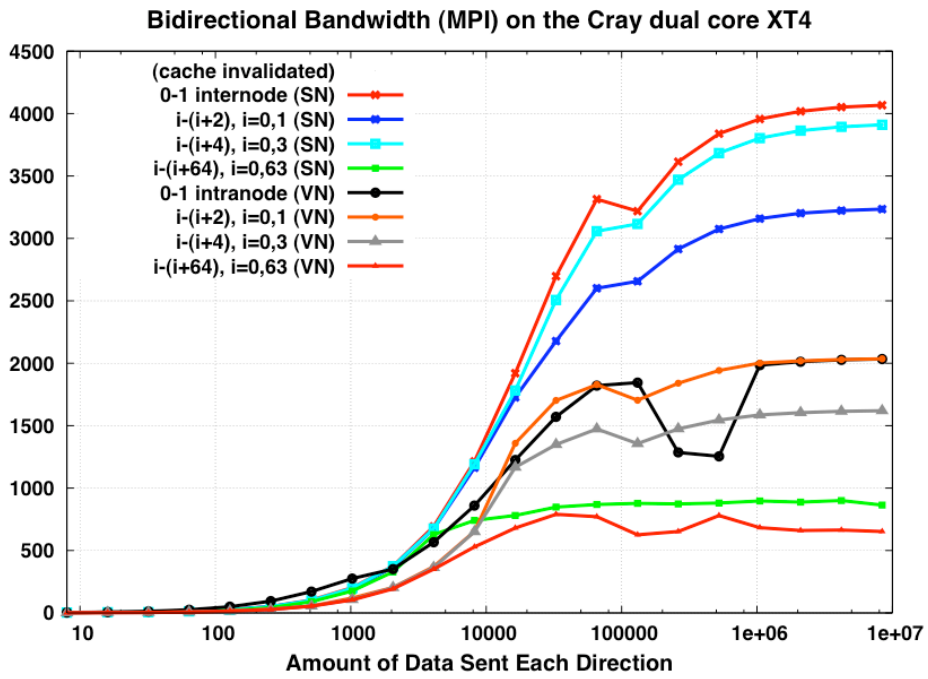
SWAP Benchmark on XT4



Same data as previous slide, plotted on a log-log scale. Note that intranode bandwidth is largest for small messages, and that simultaneous exchanges have similar bandwidths (different for SN and for VN) for smaller message sizes.

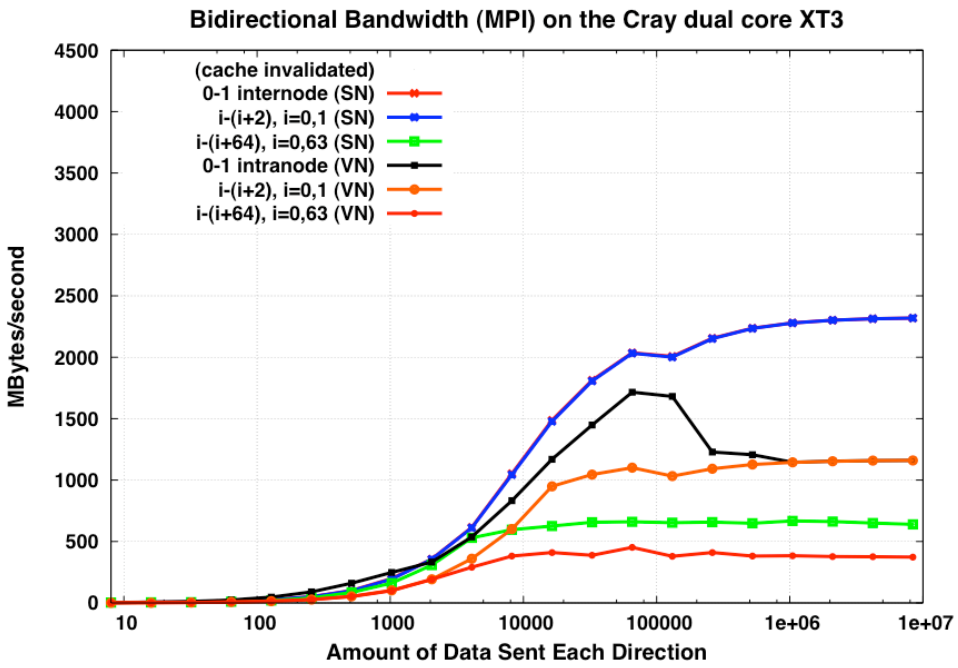
SWAP Benchmark on XT4

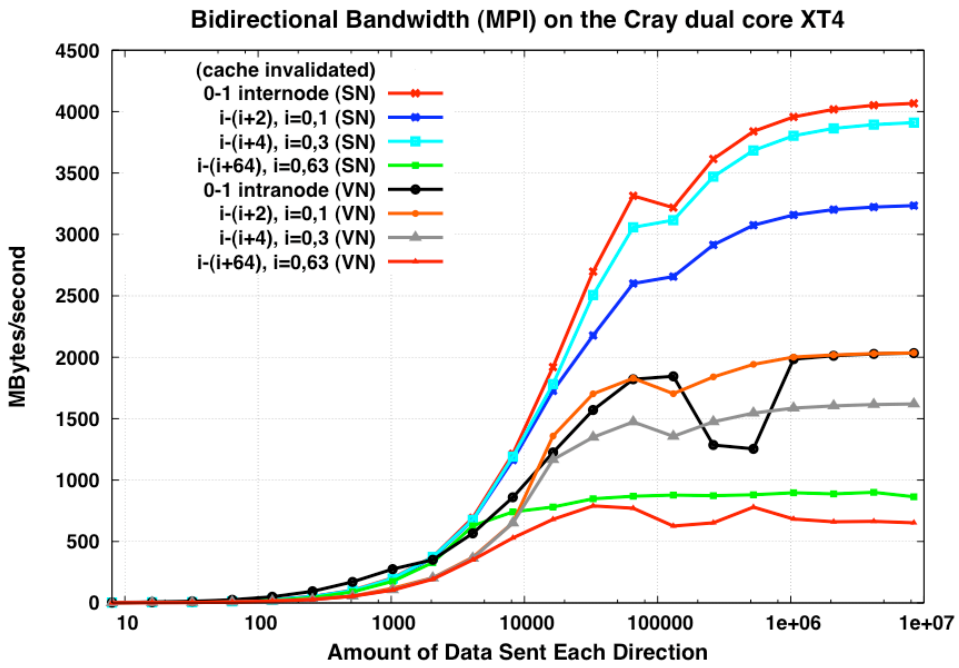




Comparing communication bandwidth with and without cache invalidation. Note that only significant difference is in intranode bandwidth. Performance is independent of cache invalidation for all experiments for small message sizes.

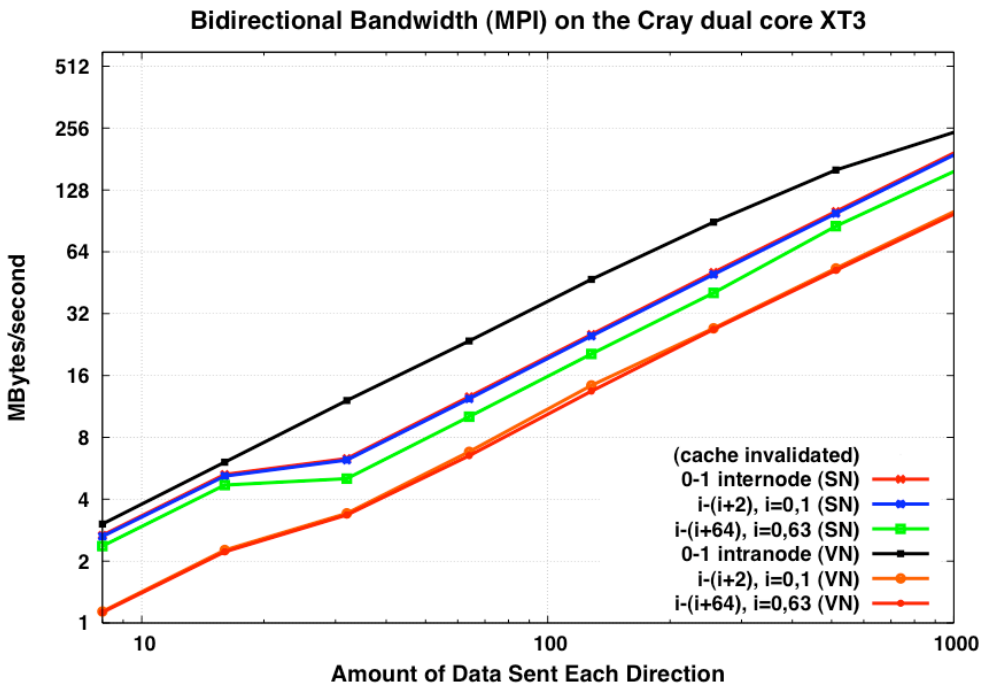
SWAP Benchmark: XT3 vs. XT4

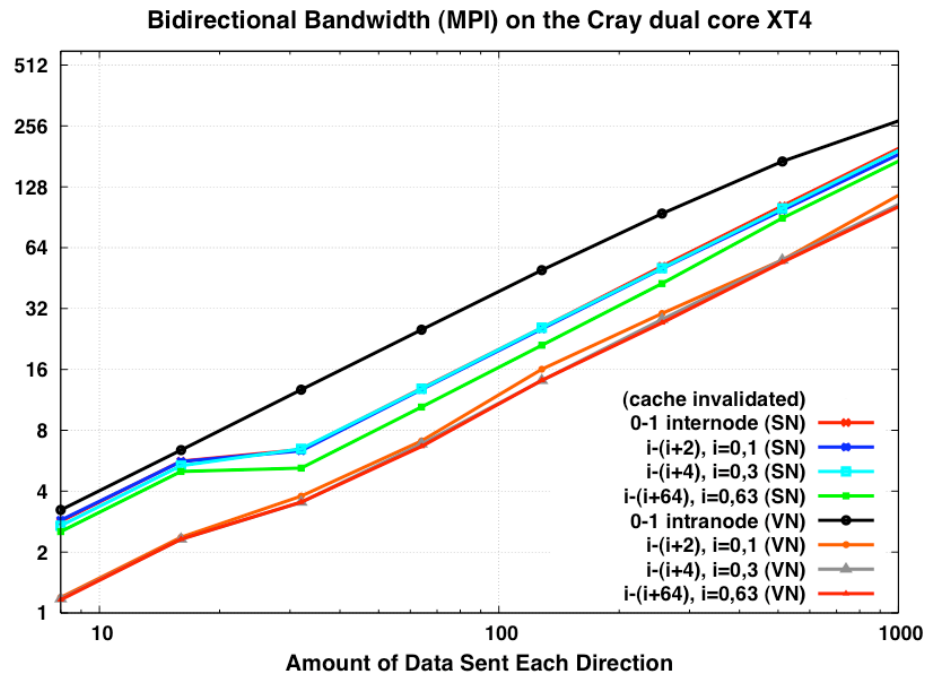




Bidirectional bandwidth on the XT4 is as much as twice that on the XT3, as advertised. The improvement is less in the situations exhibiting network link contention.

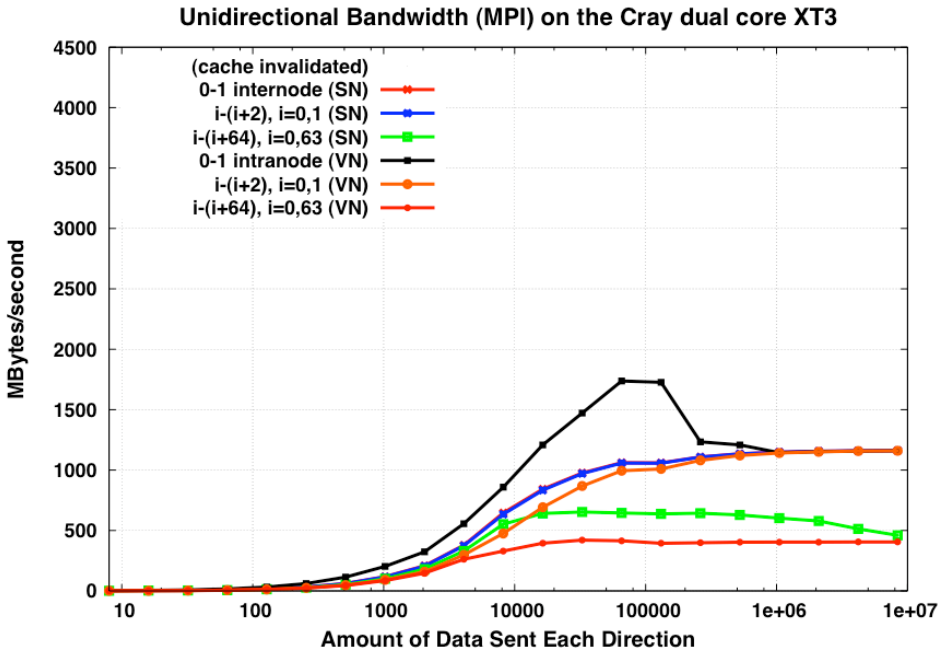
SWAP Benchmark: XT3 vs. XT4

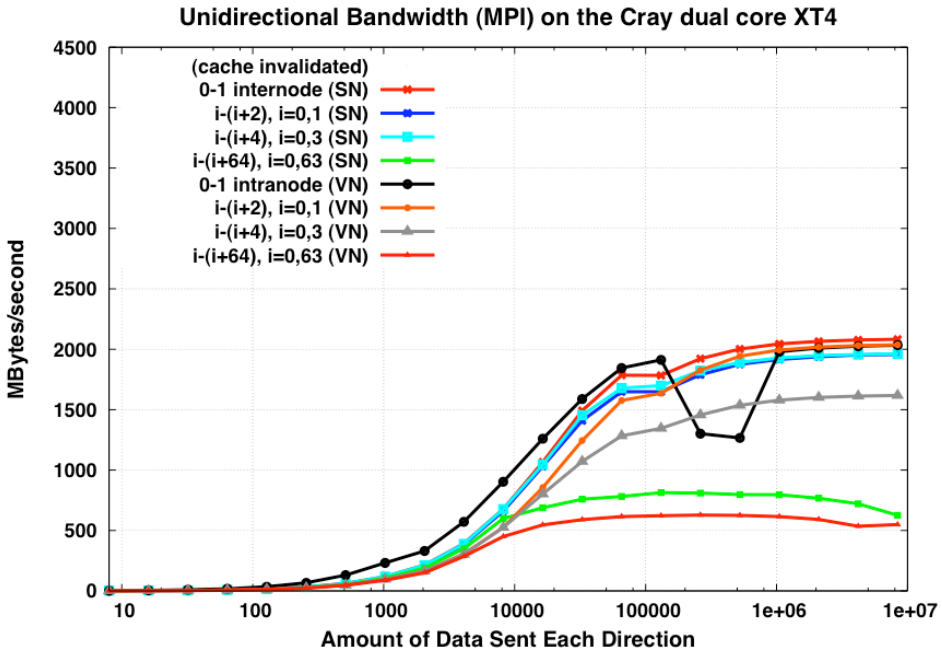




For small messages, XT3 and XT4 bidirectional communication performance are similar

SWAP Benchmark: XT3 vs. XT4





Unidirectional communication is approximately half that of bidirectional for small numbers of pairs (except intranode, where it is identical). For large numbers of pairs the improvement is less. XT3 performance is (again) half that of the XT4 when network link contention is not significant. SN/VN performance differences are less significant.

Communication Experiment Comments

1. XT4 experimental results were unchanged by use of the MPICH_FAST_MEMCPY environment variable.
2. Higher MPI bandwidth was observed on the XT4 compared to the XT3, but link contention (?) can limit the improvement.
3. Topology matters, but is complicated to exploit even if have control over process assignment. Without such control, little can be done by user (with exception of MPICH_RANK_REORDER_METHOD).
4. Optimal MPI protocols vary between experiment and message sizes:
   a) For single pair (0-1):
      i. large messages, performance is relatively insensitive to protocol.
      ii. small messages, best protocol is isend/recv for SN and sendrecv or VN.
   b) For 64 pairs simultaneously:
      i. large messages, performance is optimized by preposting receives and using ready sends (both SN and VN).
      ii. small messages, sendrecv is competitive for both SN and VN.

PSTSWM Description

The Parallel Spectral Transform Shallow Water Model represents an important computational kernel in spectral global atmospheric models. As 99% of the floating-point operations are multiply or add, it runs well on systems optimized for these operations. PSTSWM exhibits little reuse of operands as it sweeps through the field arrays; thus it exercises the memory subsystem as the problem size is scaled and can be used to evaluate the impact of memory contention in SMP nodes. PSTWM is also a parallel algorithm testbed, and all array sizes and loop bounds are determined at runtime.

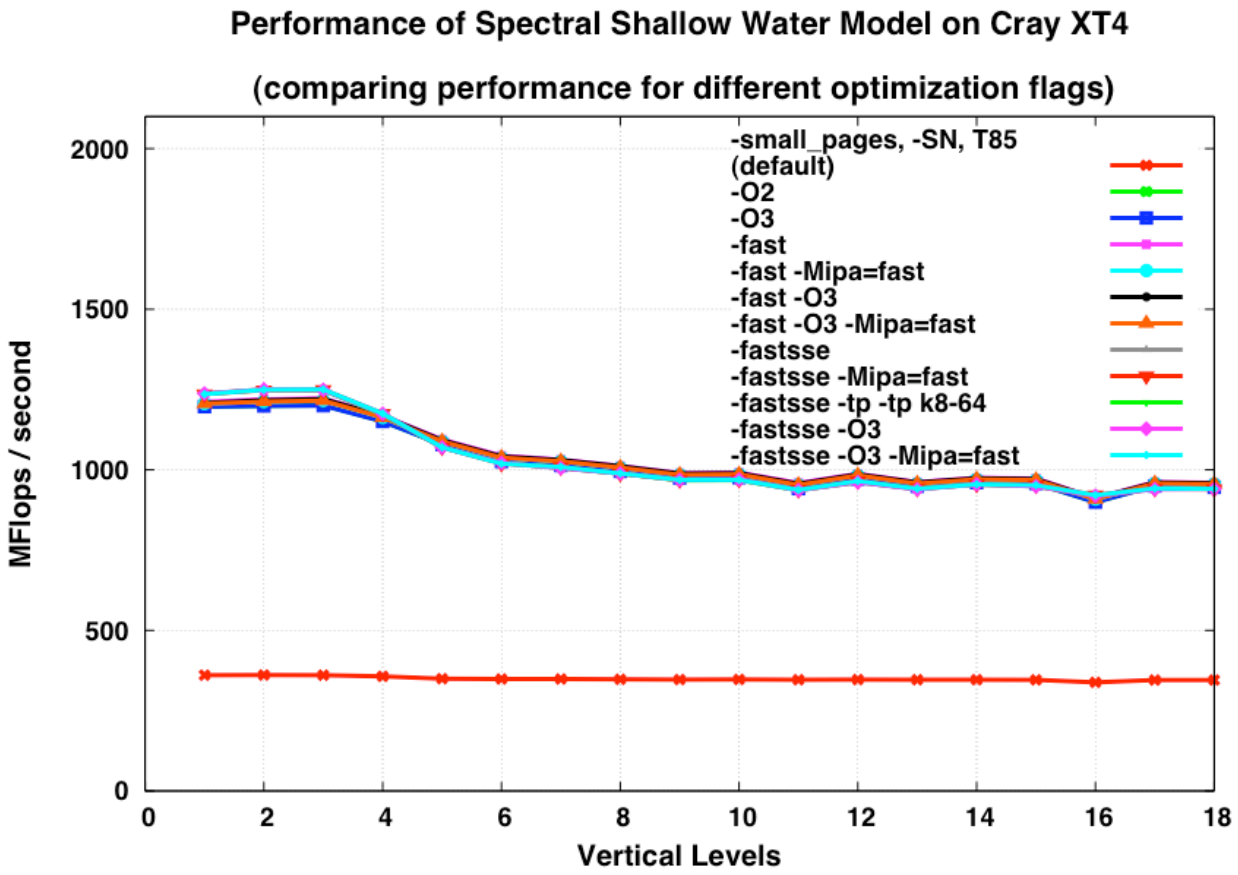
PSTSWM Experiment Particulars

These experiments examine serial performance, both using one processor and running the serial benchmark on multiple processors simultaneously. Performance is measured for a range of horizontal problems resolutions for 1 to 92 vertical levels.

Horizontal Resolutions

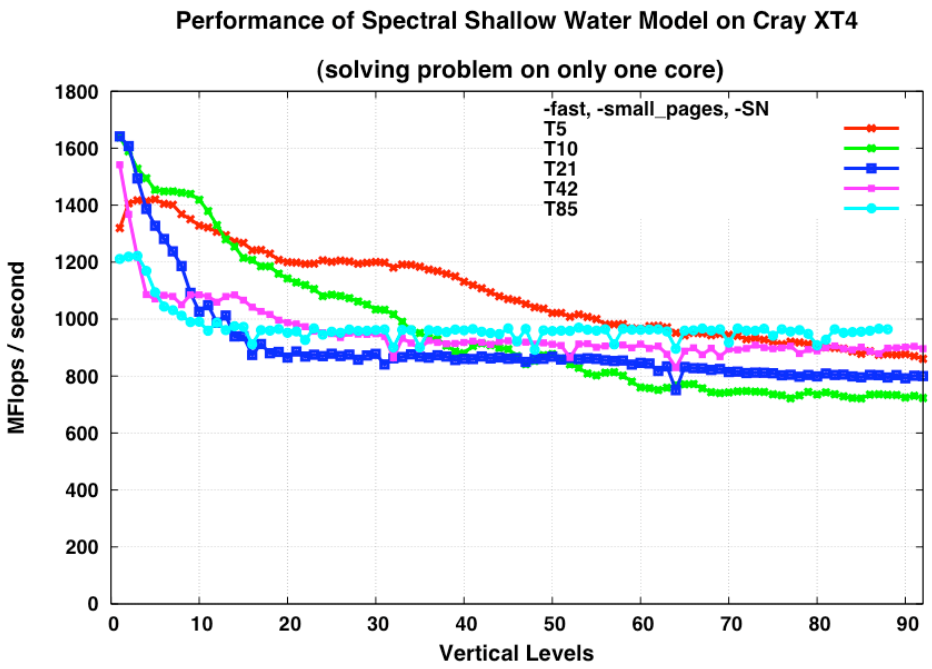
T5:	**8 x 16**
T10:	**16 x 32**
T21:	**32 x 64**
T42:	**64 x 128**
T85:	**128 x 256**
T170:	**256 x 512**
T340:	**512 x 1024**
T680:	**1024 x 2048**
T1279:	**1920 x 3840**

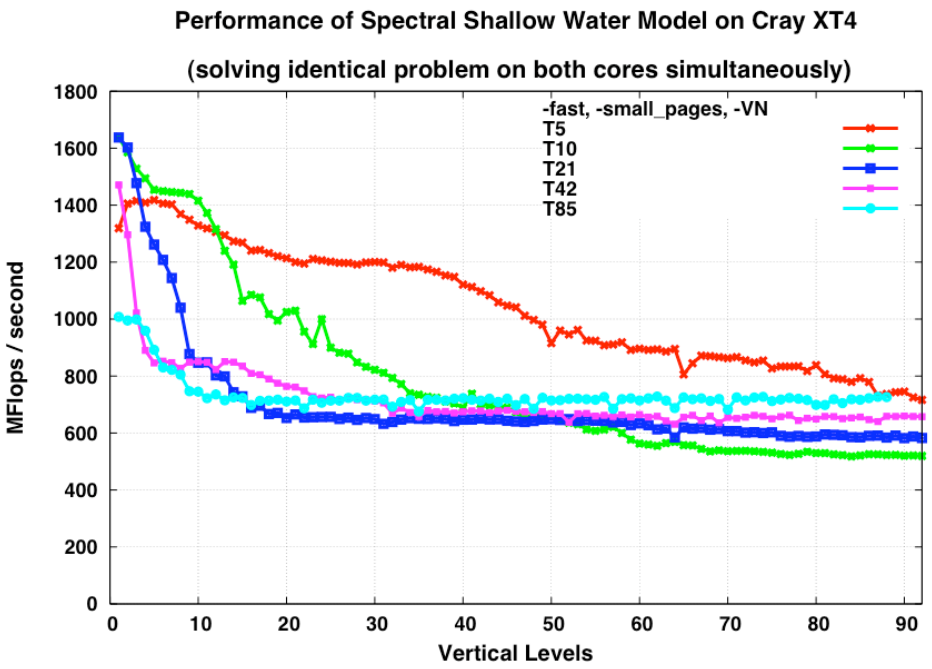
PSTSWM Compiler Comparisons on XT4



Comparing performance of different optimization levels for T85. For this code, “-fast” is as good as anything else. -small_pages also does not affect performance significantly for this code.

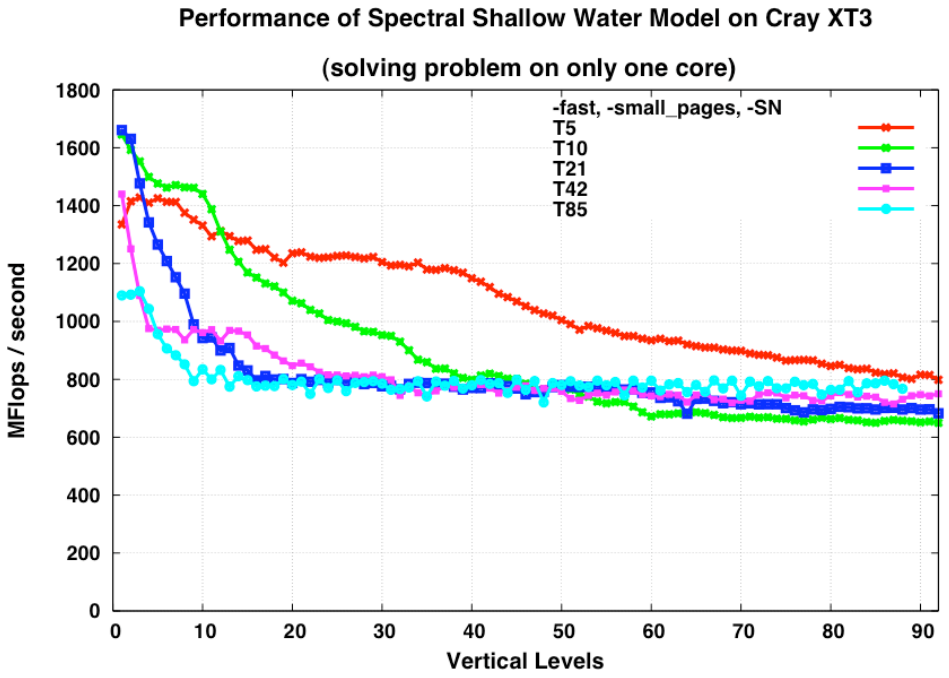
PSTSWM Performance: SN vs. VN on the XT4

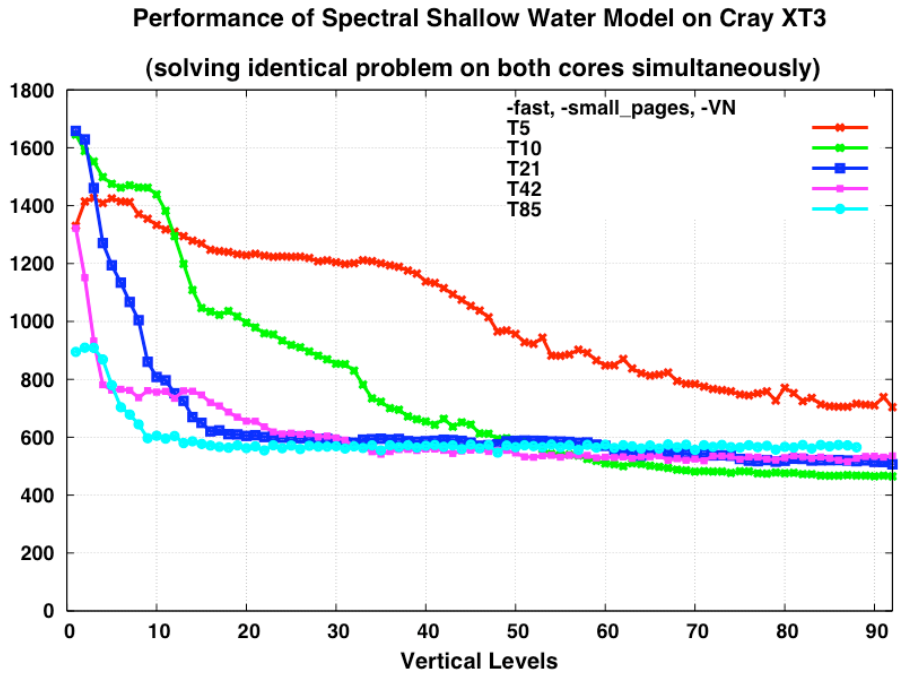




Memory contention on the XT4 degrades T85 performance by 20% for small numbers of vertical levels, increasing to 33% for large numbers. T10, T21, and T42 show little degradation for 1 level, but 35-40% degradation for 92 levels.

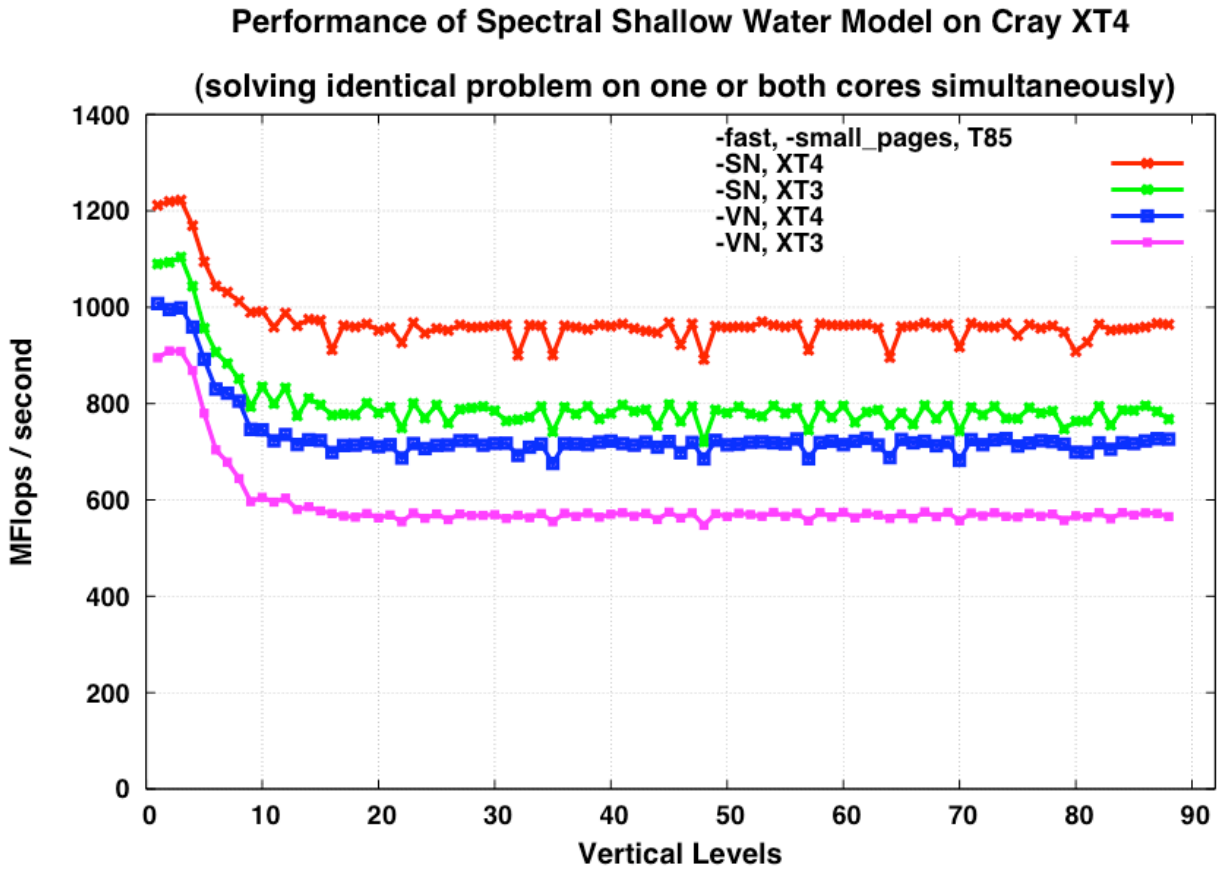
PSTSWM Performance: SN vs. VN on the XT3





Effect of memory contention on XT3 performance is qualitatively similar to that on the XT4. However, for large numbers of vertical levels, both single and dual core performance is superior on the XT4.

PSTSWM Performance: XT3 vs. XT4



Performance comparison for T85 for a range of vertical levels: improved memory performance leads to a significant performance improvement for this application, with similar improvements both with and without memory contention between the cores.

Communication Experiment Comments

1. Faster memory on the XT4 improves performance compared to the XT3, whether using one or both cores.
2. Memory contention can still be a significant performance issue when using both cores.
3. “-fast” appears to be as good as most other global options, currently. (Similar results seen for other codes, however have not tried detailed prefetching options. See Wasserman talk.)

Parallel Ocean Program (POP)

- Developed at Los Alamos National Laboratory. Used for high resolution studies and as the ocean component in the Community Climate System Model (CCSM)
- Two primary computational phases
  - Baroclinic: 3D with limited nearest-neighbor communication; scales well.
  - Barotropic: dominated by solution of 2D implicit system using conjugate gradient solves; scales poorly.
- Domain decomposition determined by grid size and 2D virtual processor grid.

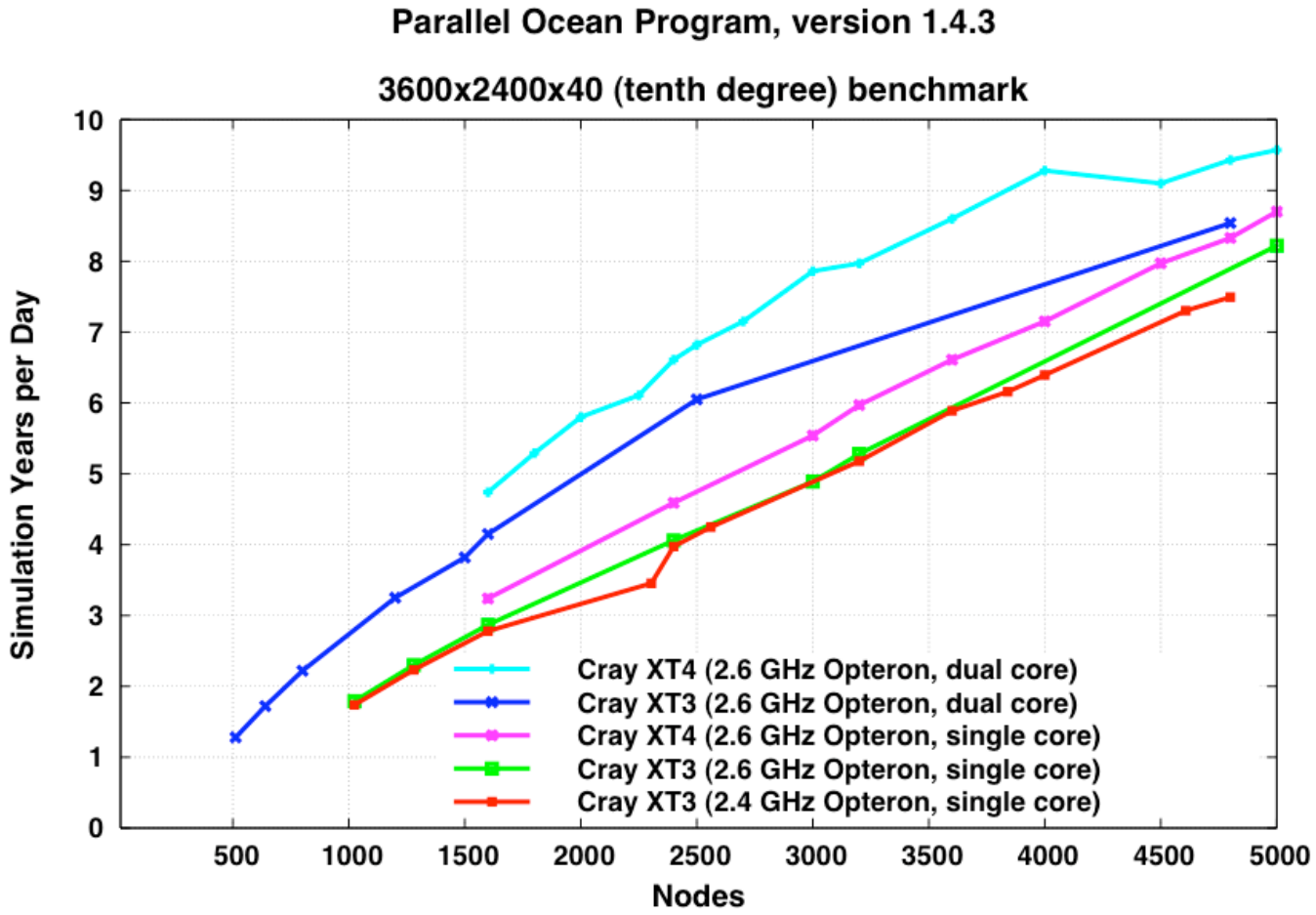
POP Experiment Particulars

- Los Alamos National Laboratory version of POP1.4.3 with a few additional parallel algorithm tuning options (due to Dr. Yoshida of CRIEPI).
- Two fixed size benchmark problems
  - Tenth degree horizontal grid of size 3600x2400x40 using internally generated horizontal grid
  - Tenth degree horizontal grid of size 3600x2400x40 using real grid

  all with very little I/O.
- Results for a given processor count are the best observed over all applicable processor grids.

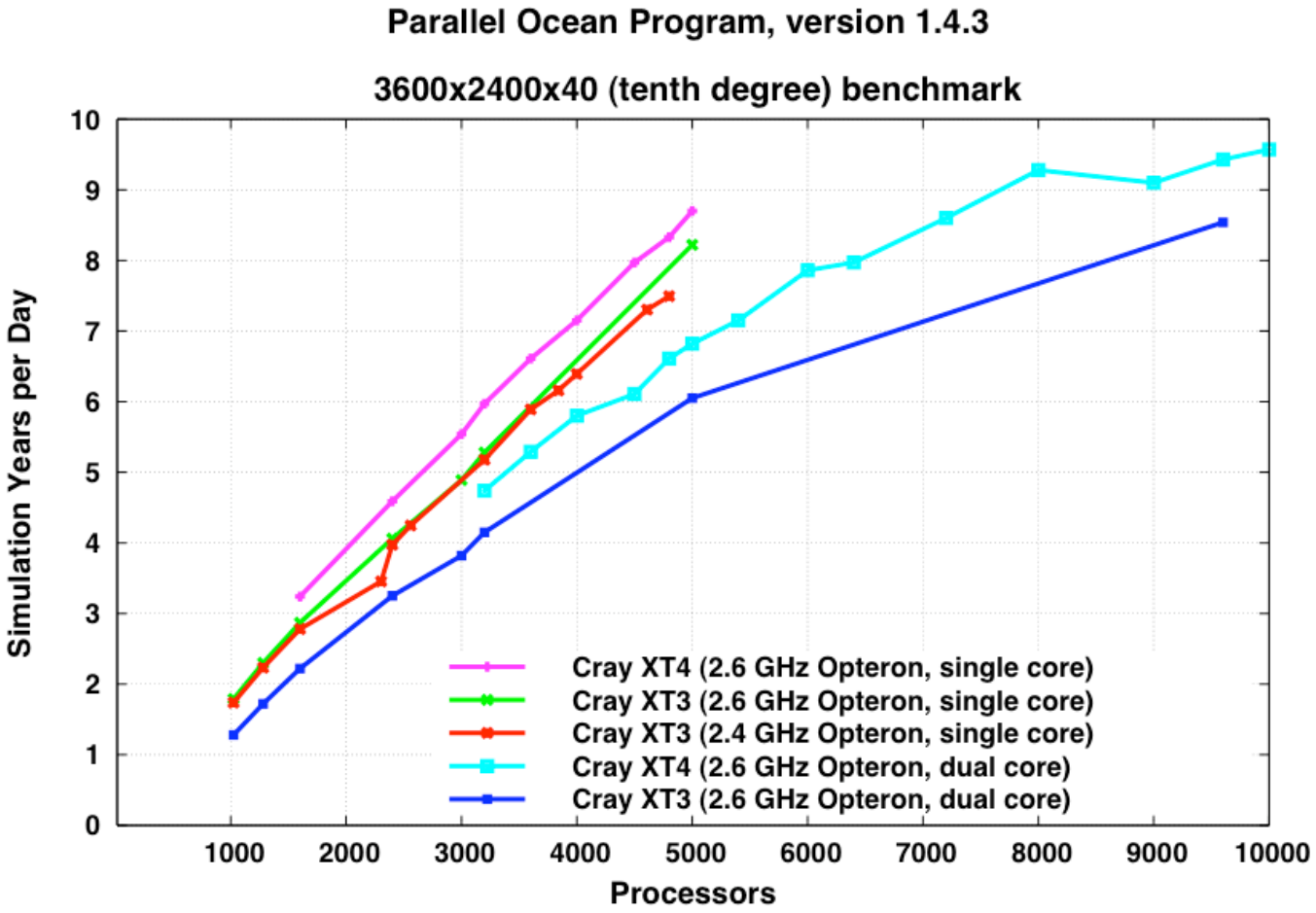
Note: This is the “original” POP benchmark. The current production version of POP is version 2.0.1, and it is the focus of current optimization work. Version 1.4.3 is being used to evaluate machine performance, not to evaluate the performance of POP.

POP Performance: XT3 vs. XT4



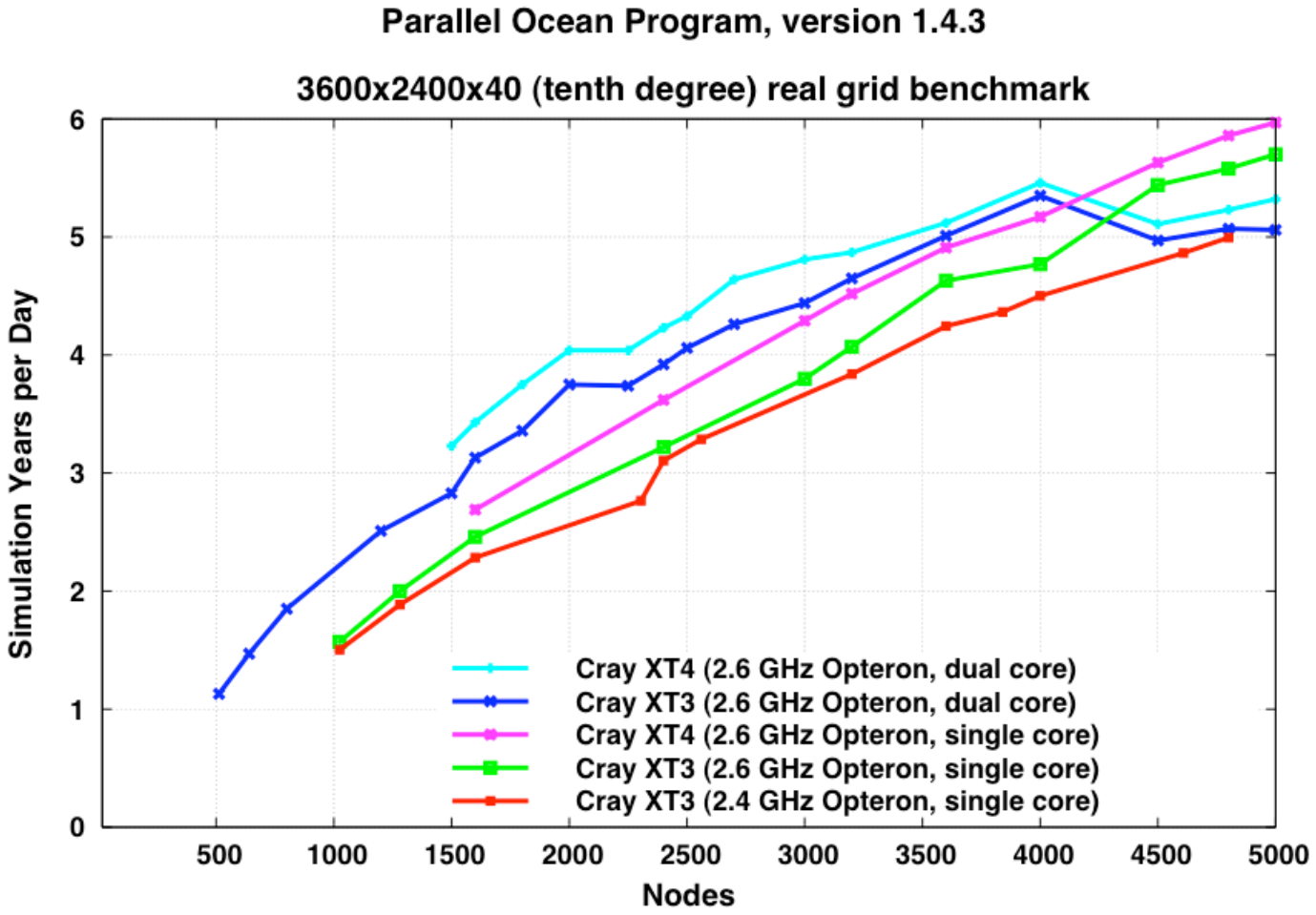
Comparison by number of nodes, so -SN using half as many cores. Good news: XT4 performance 10%-15% faster than XT3 for -VN, and 5%-10% for -SN; -VN improves performance compared to -SN for same number of nodes.

POP Performance: XT3 vs. XT4



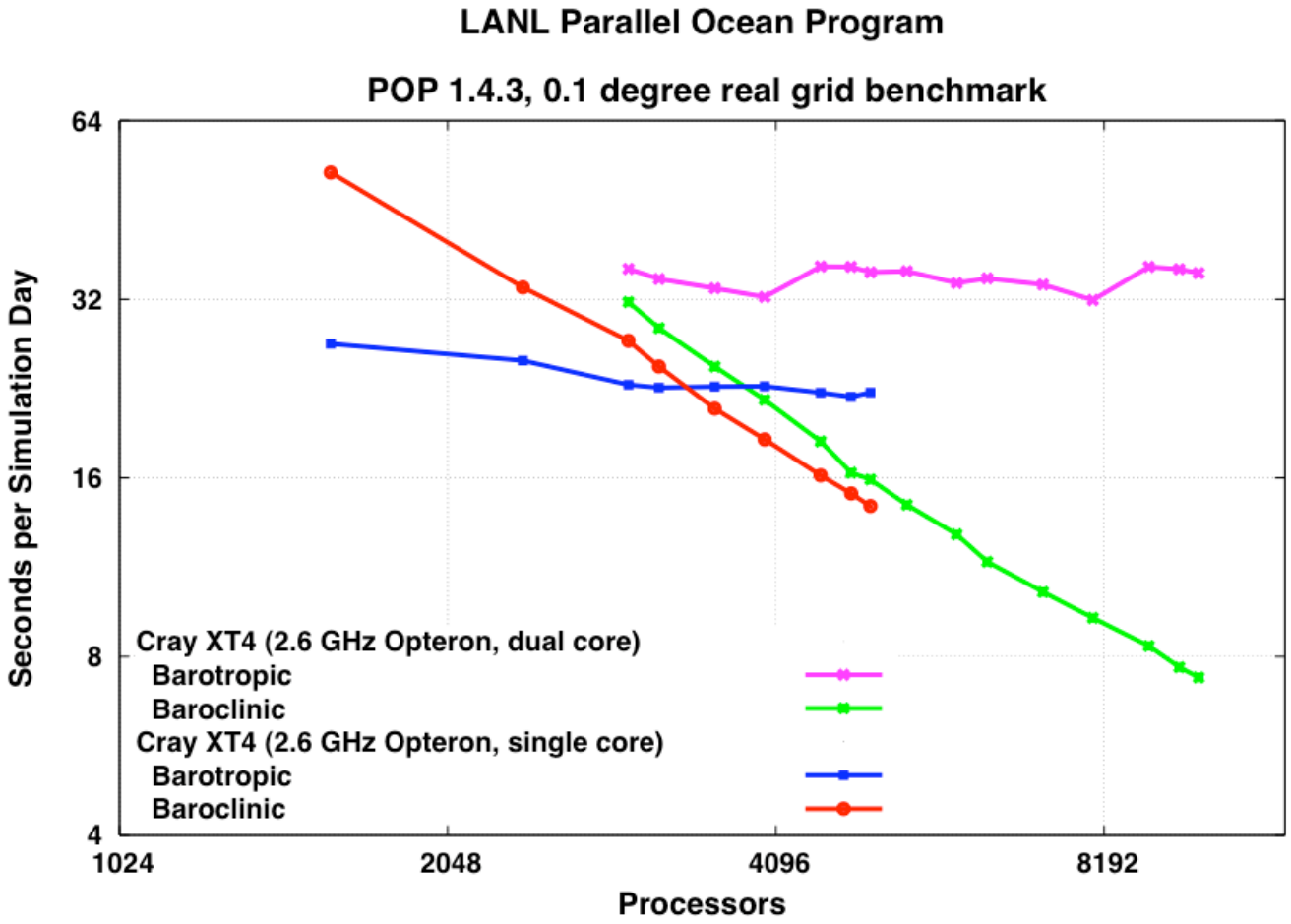
Comparison by number of cores, so -SN using twice as many nodes. Bad news: -SN performance significantly better than -VN for same number of processes (approx. 30%).

POP Performance: XT3 vs. XT4



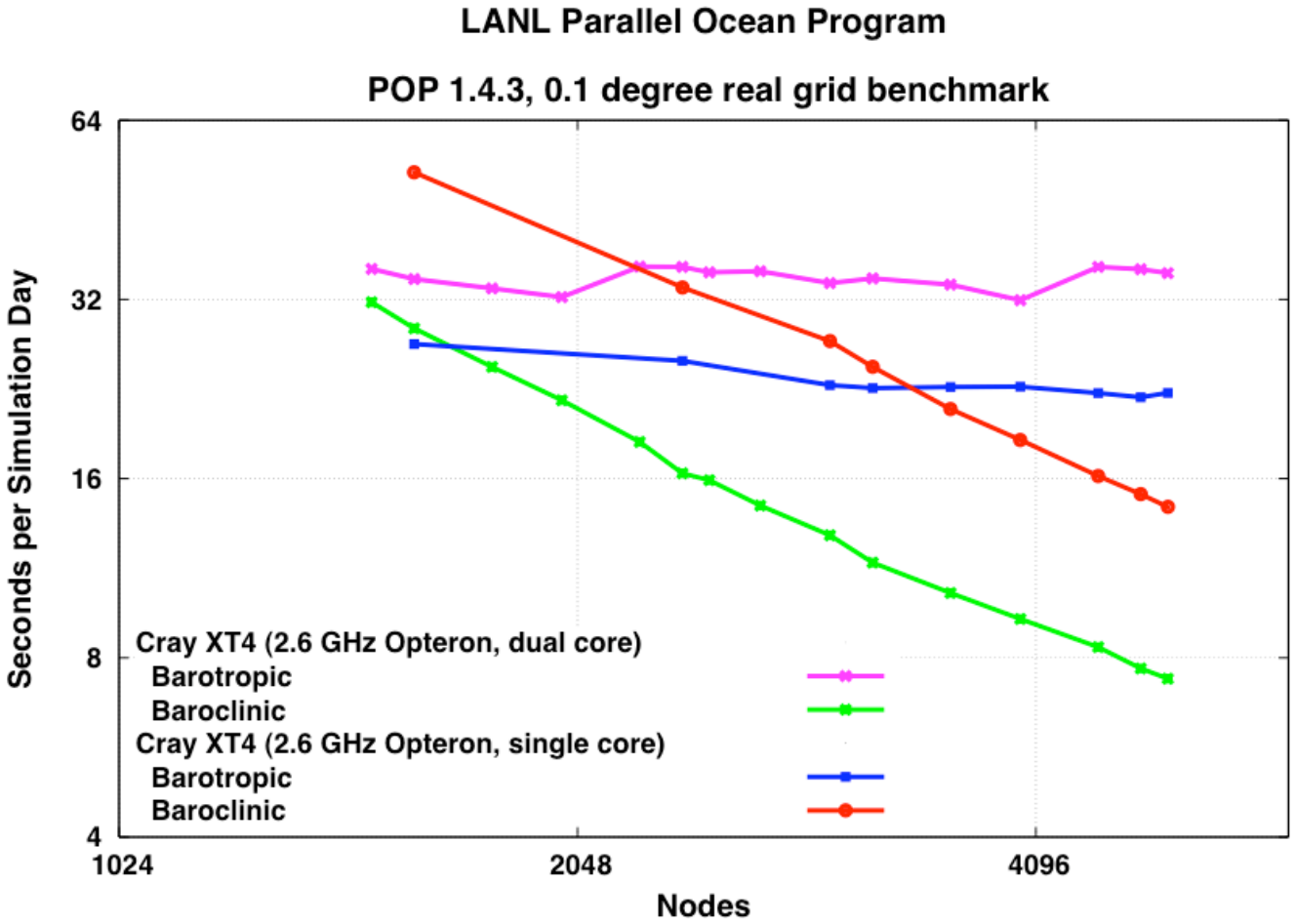
Comparison by number of nodes for real grid benchmark. More bad news: real data case increases advantage of -SN over -VN, and -SN achieves higher performance for large number of nodes, even though leaving half of the cores idle.

POP XT4 Diagnosis: -SN vs. -VN



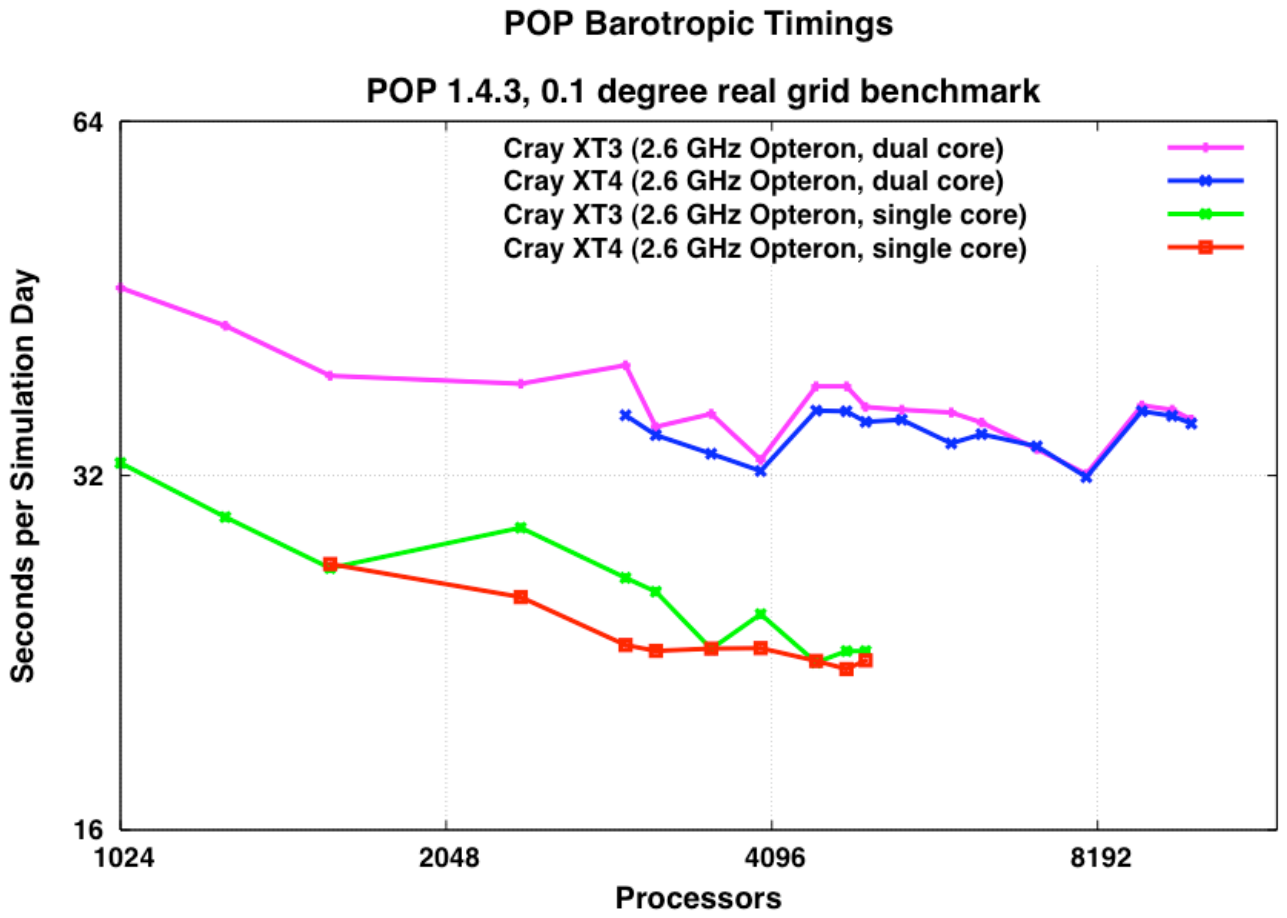
Comparison by number of cores. Baroclinic (computation-bound) is somewhat faster for single core runs, but Barotropic (allreduce-bound) is much faster for single core runs.

POP XT4 Diagnosis: -SN vs. -VN



Comparison by number of nodes. Doubling number of cores improves performance of Baroclinic significantly. Barotropic is scaling "well" in that cost is not increasing, despite being dominated by global collective communication options.

POP Barotropic Diagnosis: XT3 vs. XT4



For large processor counts, XT3 and XT4 Barotropic performance are very similar. Thus the XT4 architecture improvements did not affect Allreduce (as used in POP) performance significantly.

POP Experiment Comments

1. While not particularly sensitive to higher levels of optimization, for these experiments compiled POP with

   -O3 -fastsse -tp k8-64 -Kieee

   and ran with -small_pages and

   setenv MPICH_RANK_REORDER_METHOD 1

2. XT4 exhibits higher performance than XT3, but still does not eliminate the problem of Allreduce being much more expensive when running dual core than single core. Note that I tried different MPICH_RANK_REORDER and MPI_COLL_OPT_ON settings for selected experiments, without changing this result.

3. POP 2.0.1 (and higher) include an option to run the Barotropic on a smaller number of processors. It would be interesting to run the Barotopic on half as many processors as the Baroclinic if the subset could be distributed one per node. This might eliminate the problem.

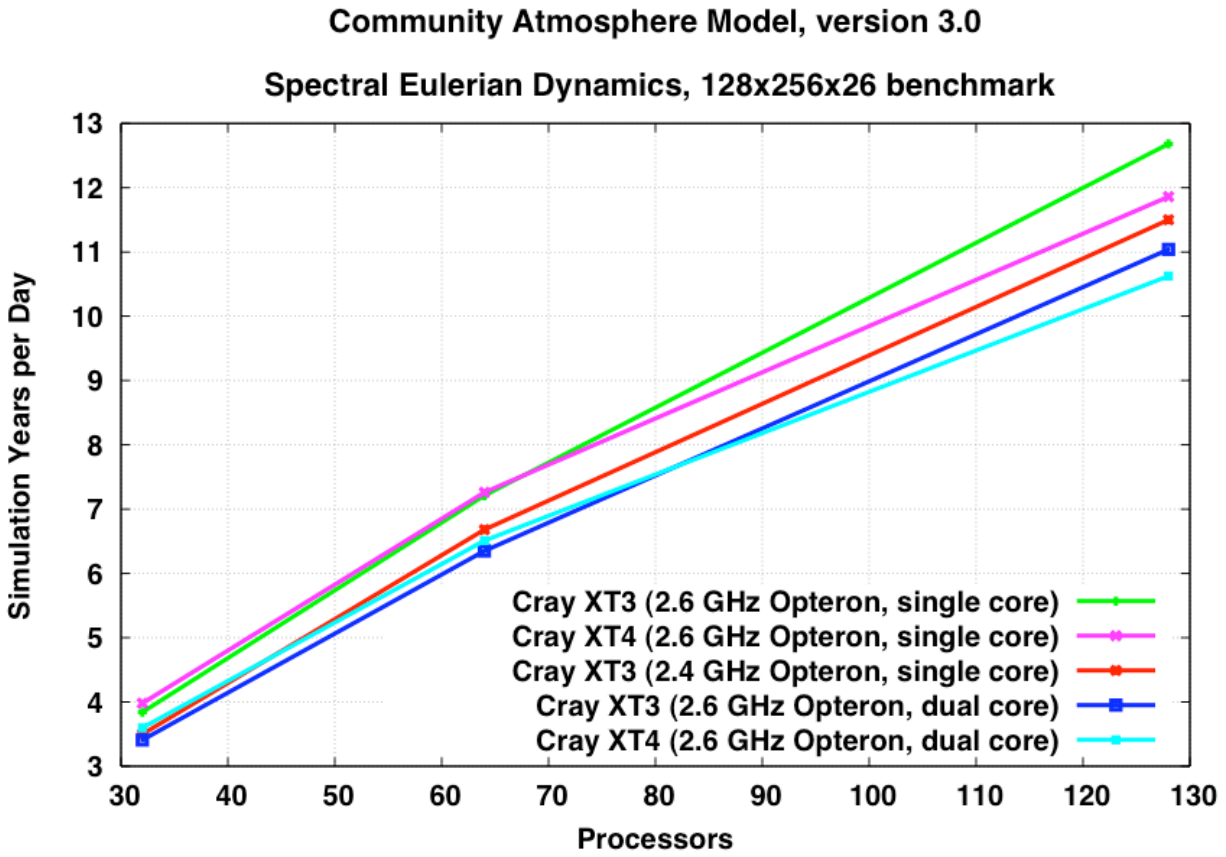
Community Atmosphere Model (CAM)

Atmospheric global circulation model

- Timestepping code with two primary phases per timestep
  - *Dynamics*: advances evolution equations for atmospheric flow
  - *Physics*: approximates subgrid phenomena, such as precipitation, clouds, radiation, turbulent mixing, …
- Multiple options for dynamics:
  - Spectral Eulerian (EUL) dynamical core (*dycore*)
  - Spectral semi-Lagrangian (SLD) dycore
  - Finite-Volume semi-Lagrangian (FV) dycore

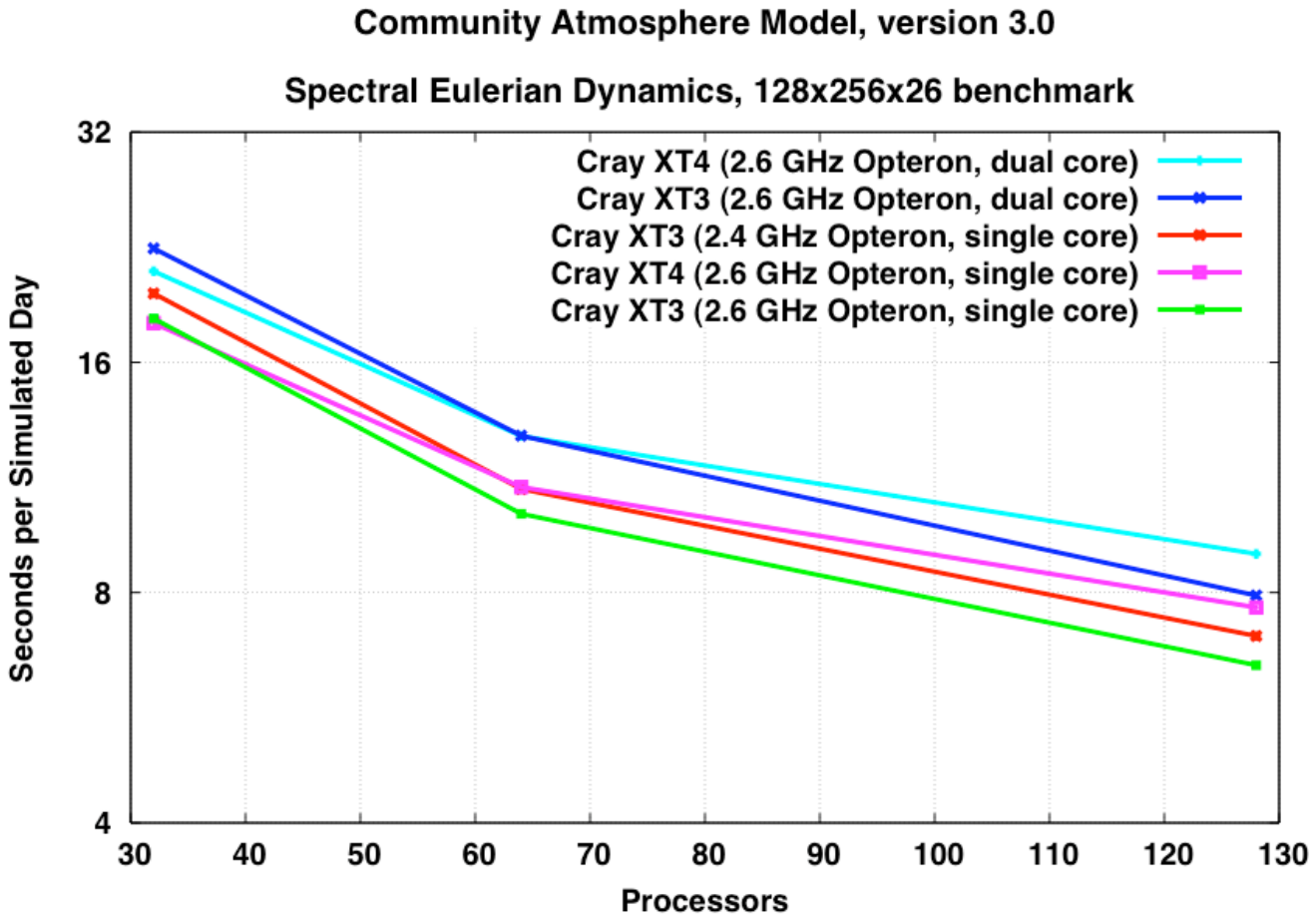
  all using tensor product *longitude* x *latitude* x *vertical level* grid over the sphere, but not same grid, same placement of variables on grid, or same domain decomposition in parallel implementation.
- Separate data structures for dynamics and physics and explicit data movement between them each timestep (in a "coupler")
- Developed at NCAR, with contributions from DOE and NASA

Spectral Eulerian Performance: XT3 vs. XT4



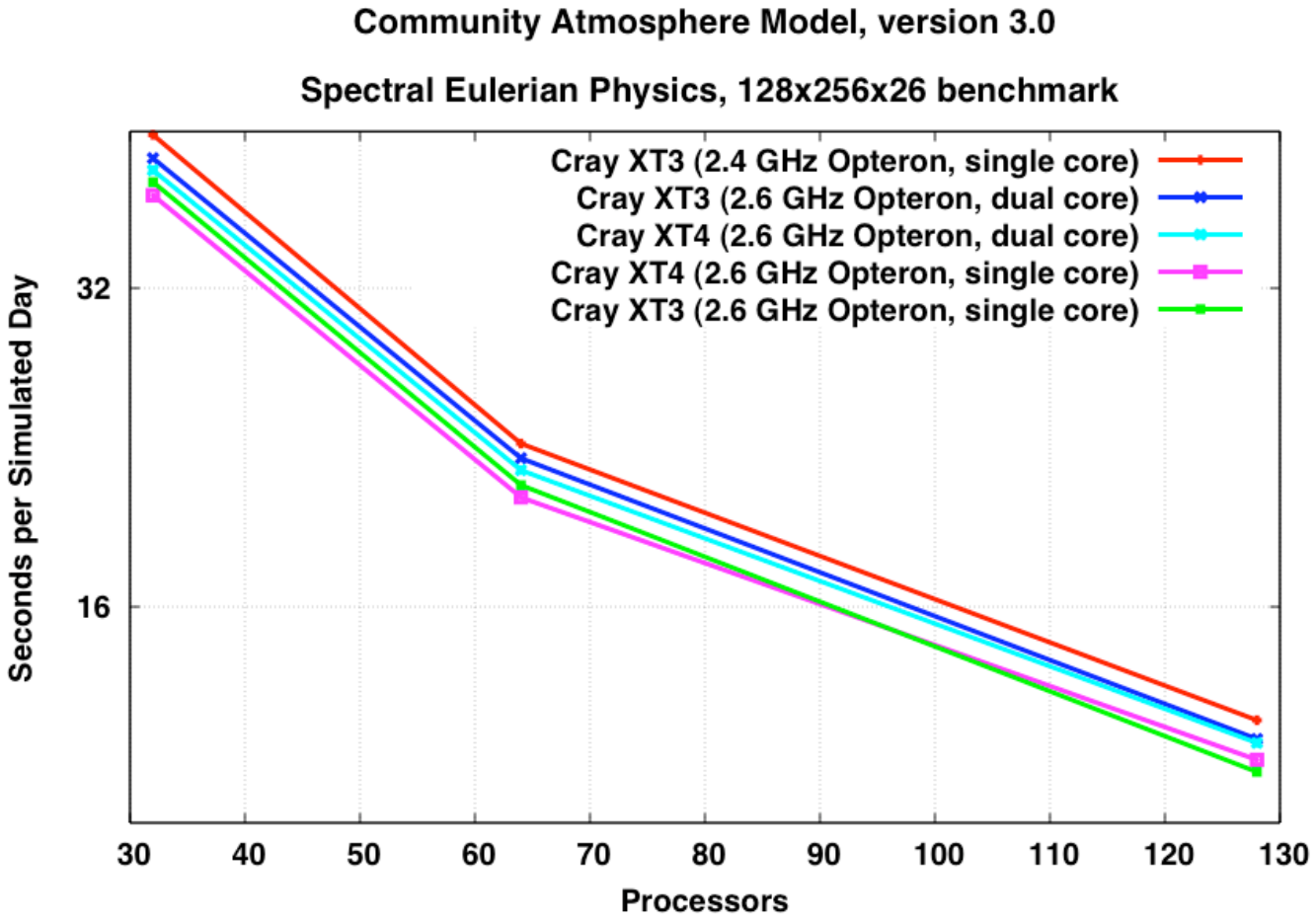
SN superior to VN by 10-15% for same process count. XT3 superior to XT4 (!!) for 128 processors. Need to rerun, but behavior of degradation is the same for both SN and VN.

Spectral Eulerian Diagnosis: XT3 vs. XT4



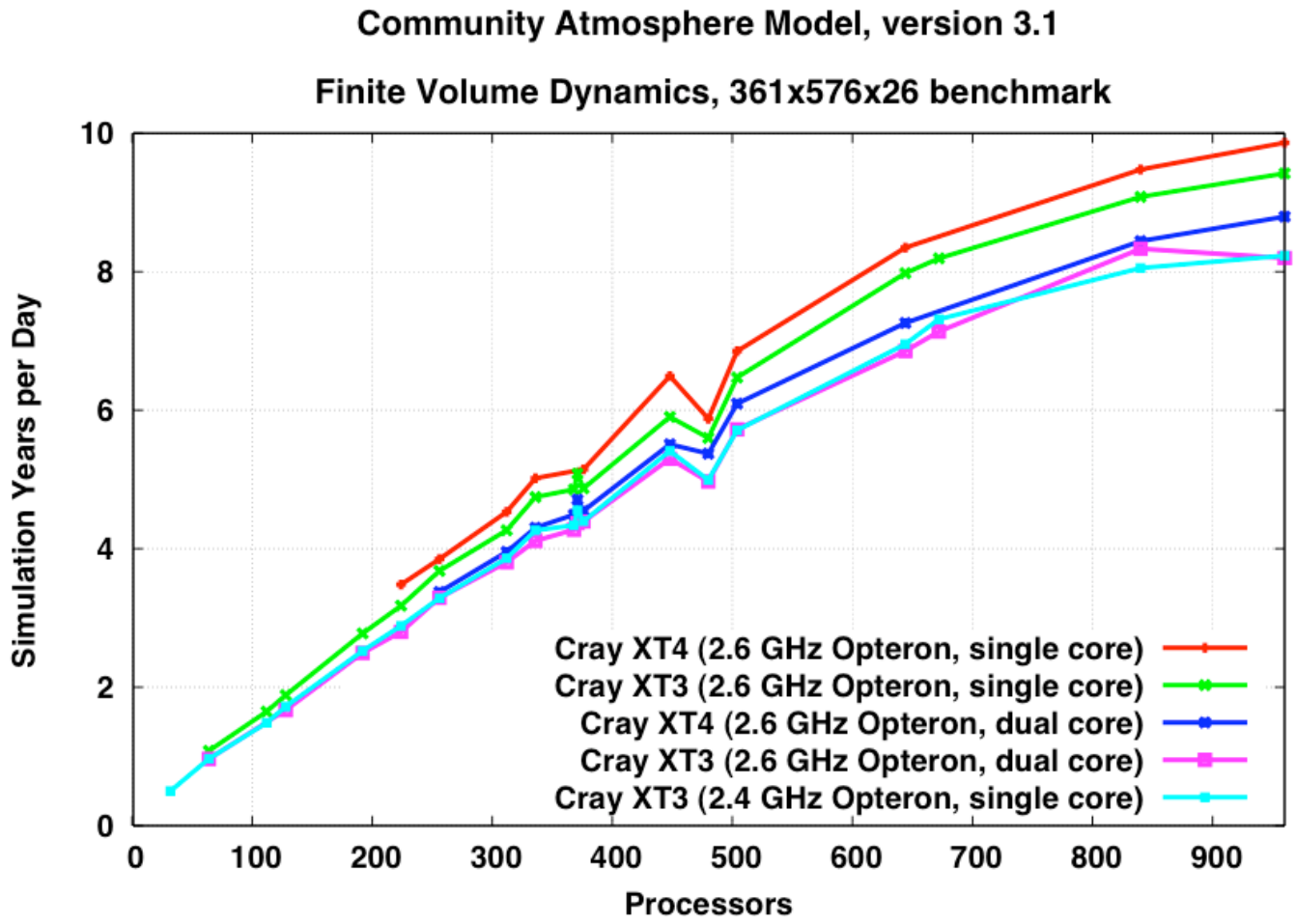
XT3 (vs. XT4) and SN (vs. VN) superiority qualitatively identical to that in dynamics alone.

Spectral Eulerian Diagnosis: XT3 vs. XT4



For physics, processor speed is most important. Single vs. dual core also has an impact, so memory contention has some importance. XT3 and XT4 have similar performance, so memory bandwidth is not important (?).

Finite Volume Performance: XT3 vs. XT4



SN performance is superior to VN performance by 10-15% for same process count. XT4 performance is 5%-10% faster than the XT3 in most cases. 2.4GHz single core XT3 has approximately same performance as 2.6GHz dual core XT3.

CAM Experiment Comments

1. Compiled CAM with -fast and ran with -small_pages and with

   setenv MPICH_RANK_REORDER_METHOD 1

2. For spectral Eulerian experiments, tried MPICH_RANK_REORDER_METHOD 1 and 2, with and without MPI_COLL_OPT_ON, and with and without MPICH_FAST_MEMCPY. Performance differences were small and within the experimental variability.

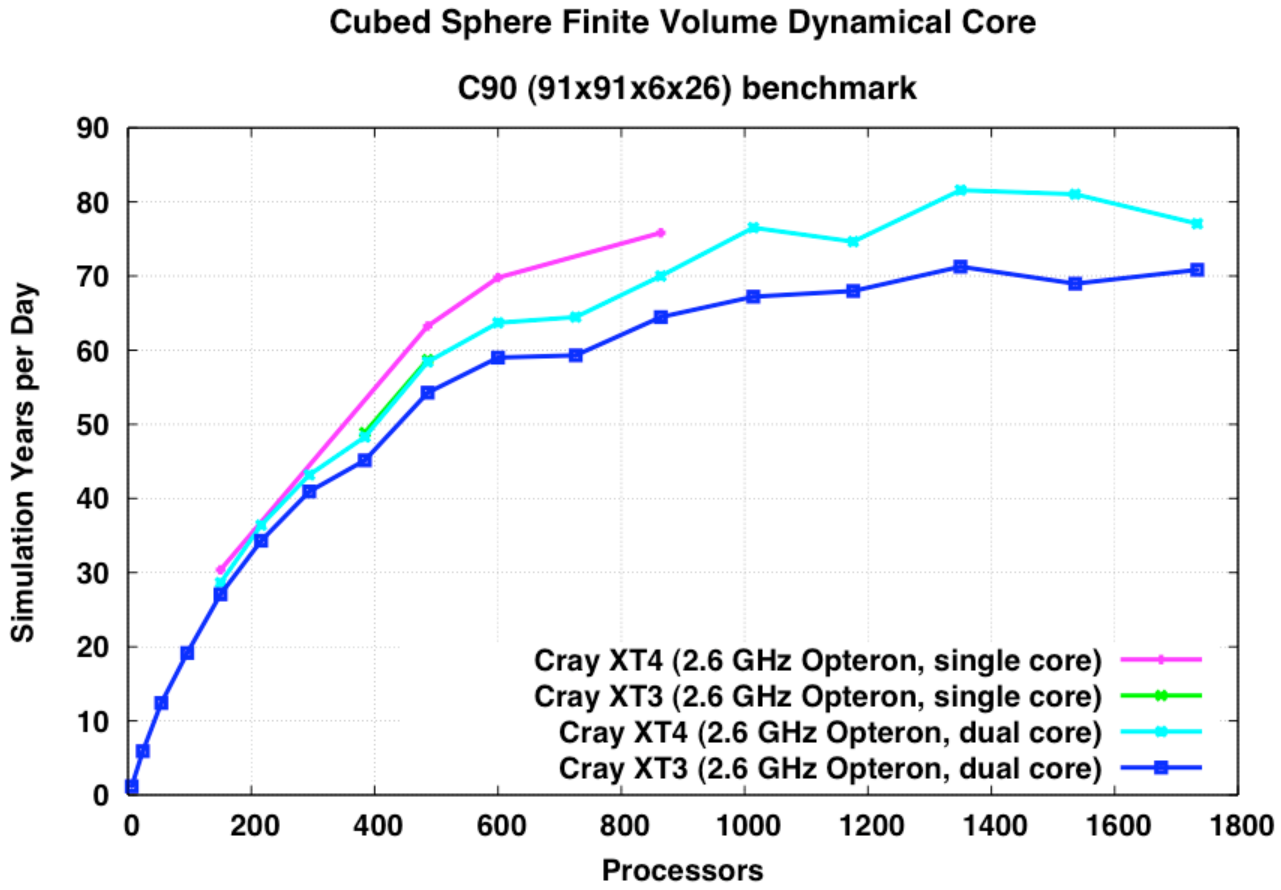
3. For finite volume dynamics, get standard results: XT4 is a little faster than the XT3, but performance is qualitatively the same. For spectral Eulerian dynamics, something is degrading XT4 performance compared to the XT3. This needs to be repeated to verify result.

Cubed Sphere Finite Volume Dynamical Core

Next generation of finite volume dynamical core currently used in CAM

- Uses cubed sphere for improved scalability (no polar filter)
- Fully explicit, using subcycling (explicit with smaller timesteps) for "fast" waves (e.g., gravity waves).
- Does not use semi-Lagrangian advection.
- Developed by S-J Lin of GFDL, William Putman of NASA Goddard, and ???. Still under development, but performance being evaluated on SGI Altix and Cray XT3/XT4.

Finite Volume Performance: XT3 vs. XT4

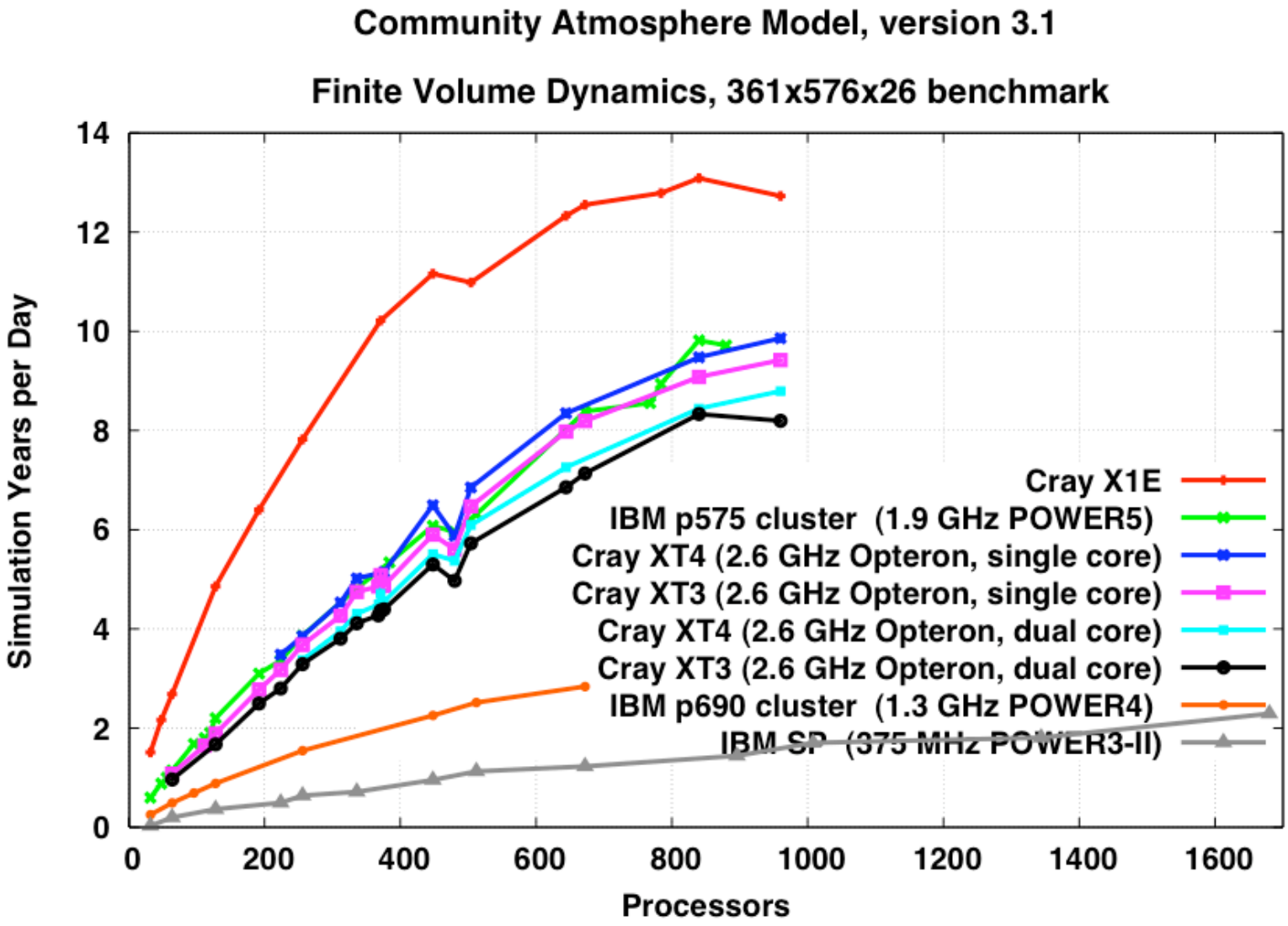


Limited data, but XT3 and XT4 performance are qualitatively similar, with XT4 faster by 5%-15%. Performance advantage of SN over VN is in the same range.

Conclusions

1. XT3/XT4 architectural differences are as advertised, with application performance (typically) higher on the XT4
2. XT3 and XT4 performance characteristics are qualitatively the same from a user's perspective.
3. The POP Allreduce single vs. dual core performance anomaly persists.
4. For most codes, single core performance is modestly better than dual core performance (for a fixed number of processes), but not enough to offset the gain from doubling the number of cores when fixing the number of nodes.
5. New environment variables did not impact performance of application codes examined in this study. (In other studies, `MPICH_RANK_REORDER_METHOD` has been important. None of the example application codes relies on Alltoall performance, so have not yet fully examined the impact of `MPI_COLL_OPT_ON`.)
6. Application code performance has not been sensitive to the choice of higher levels of optimization (except negatively), but have not tried detailed prefetching options. (See Wasserman talk.)

CAM Finite Volume Platform Comparison



POP 1.4.3 Platform Comparison